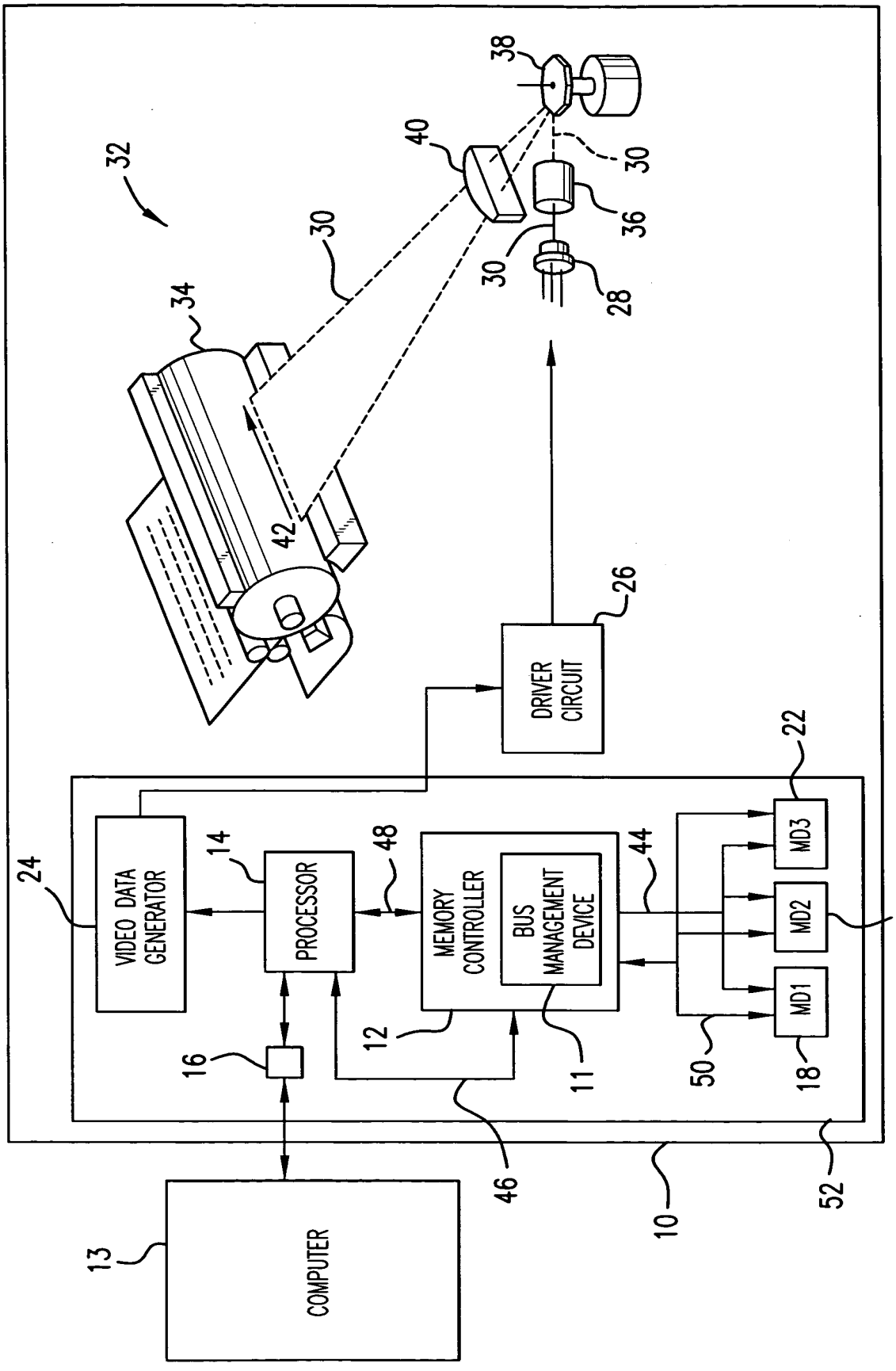

FIG. 1

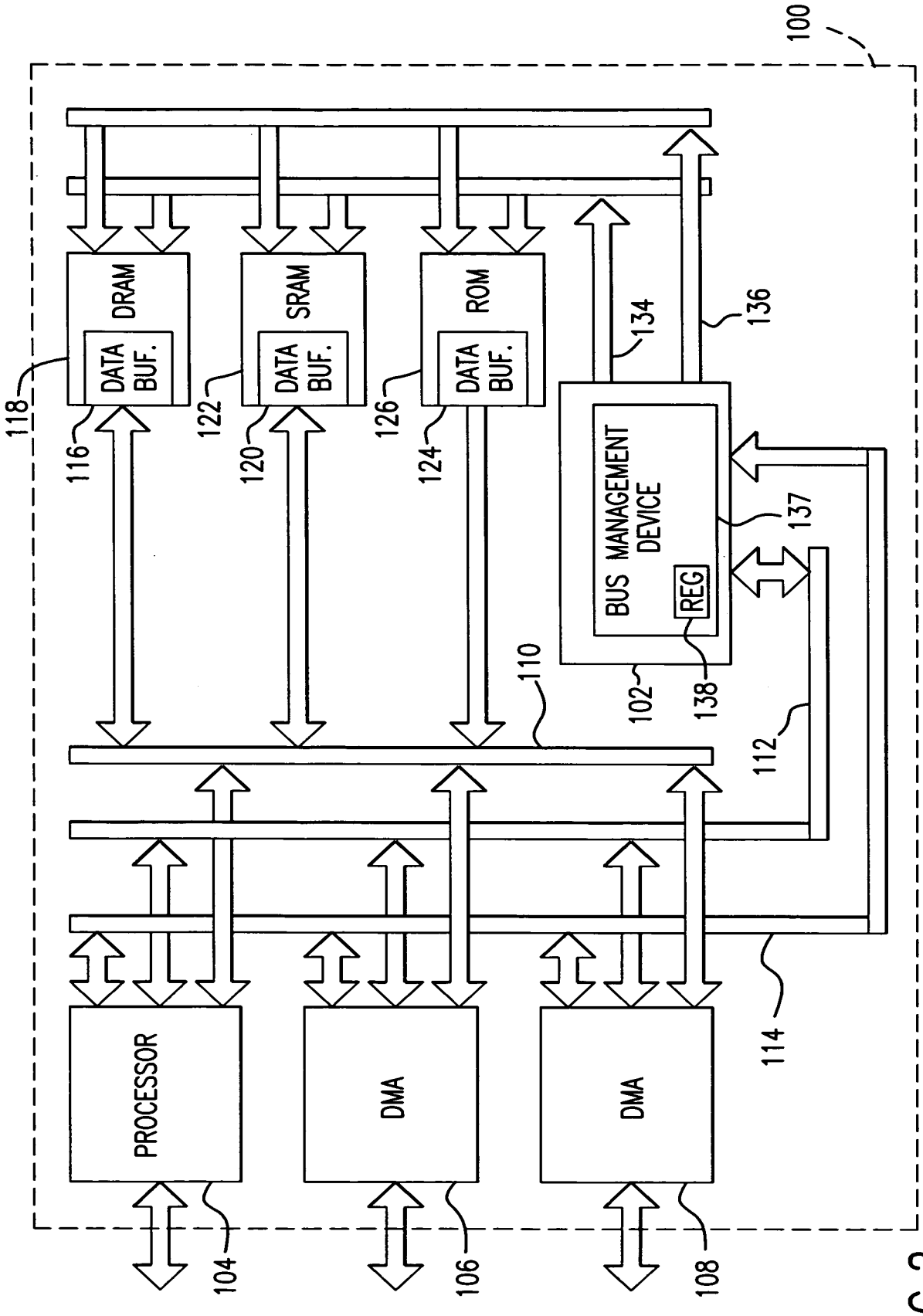
100
DRAM
DATA BUF.
SRAM
DATA BUF.
ROM
DATA BUF.
134
136
118
116
122
120
126
124
BUS MANAGEMENT DEVICE
REG
137
102
138
110
112
PROCESSOR
DMA
DMA
114
104
106
108

FIG.2

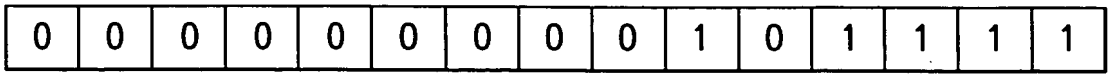

FIG.3A

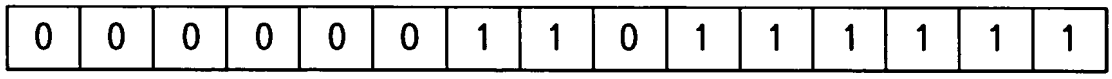

FIG.3B

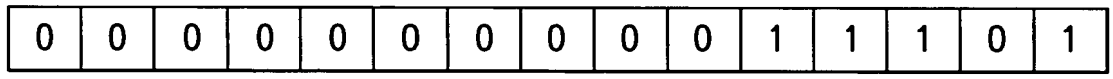

FIG.3C

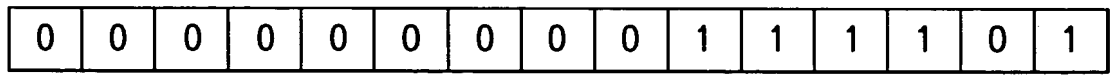

FIG.3D

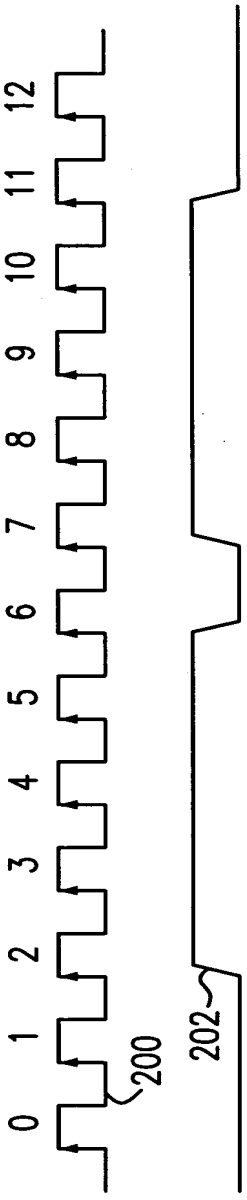

FIG.3E

	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CLOCK 0
302	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	CLOCK 1
304	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	CLOCK 2
306	0	0	0	0	0	0	1	1	1	1	0	1	1	1	1	CLOCK 3
308	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	CLOCK 4
310	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1	CLOCK 5
312	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	CLOCK 6
314	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	CLOCK 7
316	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	CLOCK 8
318	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	CLOCK 9
320	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	CLOCK 10
322	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	CLOCK 11
324	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CLOCK 12
	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	

FIG.3F

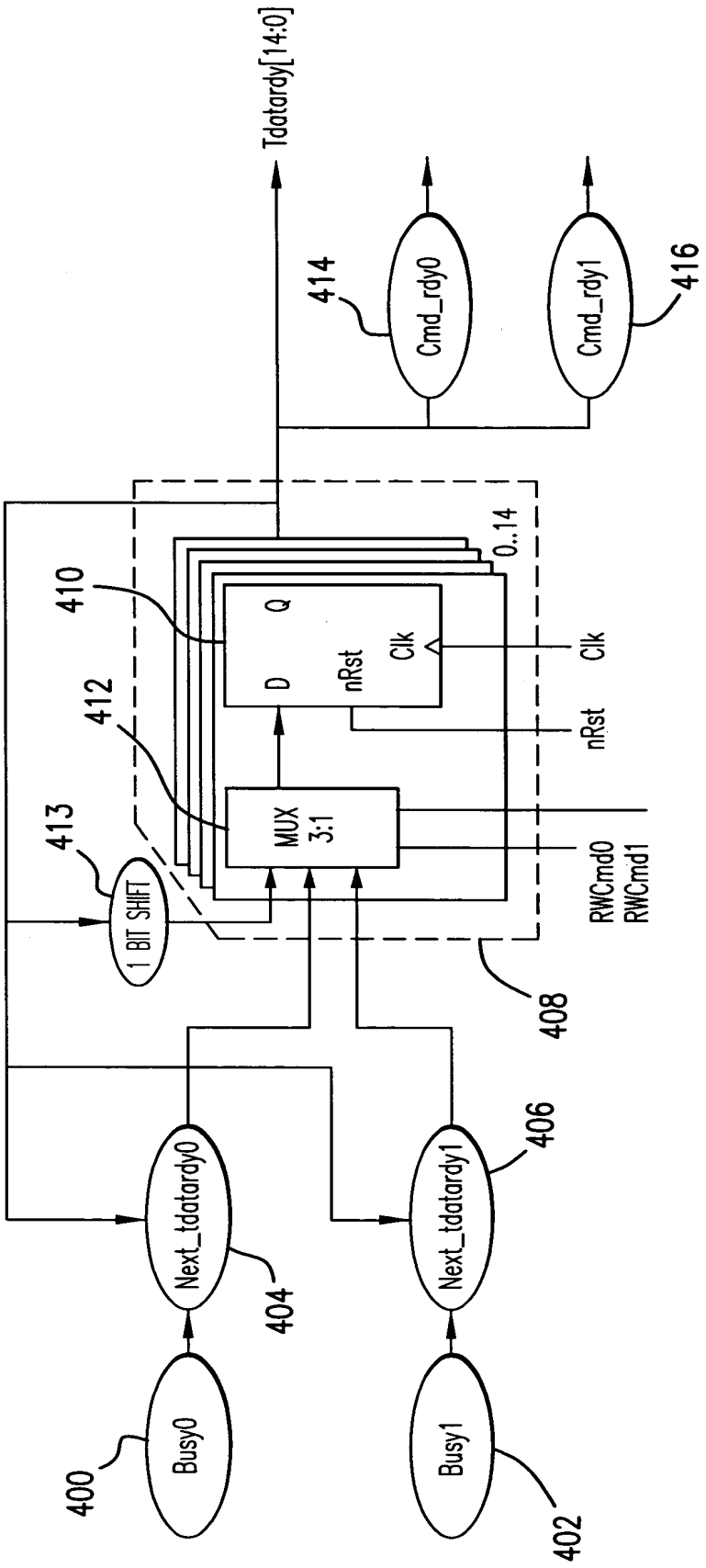
Tdatardy[14:0]
414
Cmd_rdy0
Cmd_rdy1
416
0..14
410
Q
D
nRst
Clk
Clk
nRst
412
MUX
3:1
413
1 BIT SHIFT
RWCmd0
RWCmd1
408
406
Next_tdatardy0
Next_tdatardy1
404
Busy0
Busy1
400
402

FIG.4

```
//*****************************************************************************
// SYNCHRONOUS MEMORY DATA HANDLER BLOCK
//*****************************************************************************

// Tlength0 and Tlength1: Burst length - 1.
// Len0 and Len1: Based on Tlength with a value between 0 and 3.
// Trob0 and Trob1: Rest of burst. Indicates number of words until next column address
// Busy0 and Busy1: A bit stream that represents the number of burst words.
// Tdatardy: The shift register.
// Mask: A shift register to insert wait states between accesses.
// Sync0 and Sync1: Indicates the access is for synchronous memory.
// Tread0 and Tread1: Indicates the access is a read, otherwise it is a write.
// Csd0 and Csd1: The initial access latency.
// Busy0 and Busy1: A four bit vector of 1's.
// Mask0 and Mask1: A five bit vector of 1's.
// Next_tdatardy0 and Next_tdatardy1: Value to be synchronously loaded into Tdatardy.
// Next_mask0 and Next_mask1: Value to be synchronously loaded into Mask shift register
// Cmd_rdy0 and Cmd_rdy1: Tells master that it is okay to commit to the access.
// Next_rdy0 and Next_rdy1: Logic to force a dead clock between accesses.

// ****************
// Create length that decrements based on RWCmd.
// Note: Tcnt decrements on Tdatardy in the target, which is much too late.
// ****************

    always @(Tlength0)
    begin
      if (|Tlength0[4:2])          // Length > 2'b11
       Len0 = 2'b11;
      else
       Len0 = Tlength0[1:0];
    end

// ****************
// Create length that decrements based on RWCmd.
// Note: Tcnt decrements on Tdatardy in the target, which is much too late.
// ****************

    always @(Tlength1)
    begin
      if (|Tlength1[4:2])          // Length > 2'b11
       Len1 = 2'b11;
      else
       Len1 = Tlength1[1:0];
    end
```

FIG.5A

```
// ****************
// Convert Trob into bit stream.
// ****************

     always @(Trob0 or Len0)
     begin
       case (Trob0)                    // synopsys full_case parallel_case
       2'h0:            Busy0[3:0] = 4'b0001;
       2'h1:   case (Len0[1:0])        // synopsys full_case parallel_case
               2'h0:    Busy0[3:0] = 4'b0001;
               2'h1:    Busy0[3:0] = 4'b0011;
               2'h2:    Busy0[3:0] = 4'b0011;
               2'h3:    Busy0[3:0] = 4'b0011;
               endcase
       2'h2    case  (Len0[1:0])       // synopsys full_case parallel_case
               2'h0:    Busy0[3:0] = 4'b0001;
               2'h1:    Busy0[3:0] = 4'b0011;
               2'h2:    Busy0[3:0] = 4'b0111;
               2'h3:    Busy0[3:0] = 4'b0111;
               endcase
       2'h3:   case  (Len0[1:0])       // synopsys full_case parallel_case
               2'h0:    Busy0[3:0] = 4'b0001;
               2'h1:    Busy0[3:0] = 4'b0011;
               2'h2:    Busy0[3:0] = 4'b0111;
               2'h3:    Busy0[3:0] = 4'b1111;
               endcase
       endcase
     end

// ****************
// Convert Trob into bit stream.
// ****************

     always @(Trob1 or Len1)
     begin
       case (Trob1)                    // synopsys full_case parallel_case
       2'h0:            Busy1[3:0] = 4'b0001;
       2'h1:   case (Len1[1:0])        // synopsys full_case parallel_case
               2'h0:    Busy1[3:0] = 4'b0001;
               2'h1:    Busy1[3:0] = 4'b0011;
               2'h2:    Busy1[3:0] = 4'b0011;
               2'h3:    Busy1[3:0] = 4'b0011;
               endcase
       2'h2:   case (Len1[1:0])        // synopsys full_case parallel_case
               2'h0:    Busy1[3:0] = 4'b0001;
               2'h1:    Busy1[3:0] = 4'b0011;
               2'h2:    Busy1[3:0] = 4'b0111;
               2'h3:    Busy1[3:0] = 4'b0111;
               endcase
       2'h3:   case (Len1[1:0])        // synopsys full_case parallel_case
               2'h0:    Busy1[3:0] = 4'b0001;
               2'h1:    Busy1[3:0] = 4'b0011;
               2'h2:    Busy1[3:0] = 4'b0111;
               2'h3:    Busy1[3:0] = 4'b1111;
               endcase
       endcase
     end
```

FIG.5B

```
// ****************
// Shift right by 1.  Shift in zero for MSB.
// FIX: the high order bit should be shifted in as zero in the Next_tdatardy block.
// ****************

    always @(posedge Clk or negedge nRst)
    begin
      if (!nRst)
      begin
        Tdatardy <= 15'h0000;
        Mask     <= 15'h0000;
      end
      else if (RWCmd0 & !RWCmd1)
      begin
       Tdatardy[13:0] <= Next_tdatardy0;
        Tdatardy[14]   <= 0;

       Mask[13:0]     <= Next_mask0;
       Mask[14]       <= 0;
      end
      else if (RWCmd1 & !RWCmd0)
      begin
       Tdatardy[13:0] <= Next_tdatardy1;
       Tdatardy[14]   <= 0;

       Mask[13:0]     <= Next_mask1;
       Mask[14]       <= 0;
      end
      else     // shift
      begin
       Tdatardy[13:0] <= Tdatardy[14:1];
       Tdatardy[14]   <= 0;

        Mask[13:0]     <= Mask[14:1];
        Mask[14]       <= 0;
      end
    end
```

FIG.5C

```
// ****************
// Generate next bit stream by concatenating old bit stream with new bit stream.
// Bit stream includes a right shift of one bit.
// ****************

  always @(Tdatardy or Csd0 or Busy0 or Sync0 or Tread0 or Mask0)
  begin
    if (Sync0 && Tread0)              // Read
    begin
     casex (Csd0)                     // synopsys parallel_case
     3'h1: begin
             Next_tdatardy0 = {6'h0,Busy0,Tdatardy[4:1]};
             Next_mask0     = {5'h0,Mask0,Tdatardy[4:1]};
           end
      3'h2: begin
             Next_tdatardy0 = {5'h0,Busy0,Tdatardy[5:1]};
             Next_mask0     = {4'h0,Mask0,Tdatardy[5:1]};
           end
      3'h3: begin
             Next_tdatardy0 = {4'h0,Busy0,Tdatardy[6:1]};
             Next_mask0     = {3'h0,Mask0,Tdatardy[6:1]};
           end
      3'h4: begin
             Next_tdatardy0 = {3'h0,Busy0,Tdatardy[7:1]};
             Next_mask0     = {2'h0,Mask0,Tdatardy[7:1]};
           end
      3'h5: begin
             Next_tdatardy0 = {2'h0,Busy0,Tdatardy[8:1]};
             Next_mask0     = {1'h0,Mask0,Tdatardy[8:1]};
           end
      3'h6: begin
             Next_tdatardy0 = {1'h0,Busy0,Tdatardy[9:1]};
             Next_mask0     = {Mask0,Tdatardy[9:1]};
           end
       default: begin
                  Next_tdatardy0 = 14'bxx_xxxx_xxxx_xxxx;
                  Next_mask0     = 14'bxx_xxxx_xxxx_xxxx;
                end
     endcase
    end
   else if (Sync0 && !Tread0)      // Write
    begin
     Next_tdatardy0 = {9'h00,Busy0,Tdatardy[1]};
     Next_mask0     = {8'h00,Mask0,Tdatardy[1]};
    end
   else                            // async
  begin
     Next_tdatardy0 = 14'h0000;
     Next_mask0     = 14'h0000;
    end

end
```

FIG.5D

```
// ****************
// Determine if new access can issue R/W command.
// ****************

    always @(Tdatardy or Mask or Csd1 or Sync1 or Tread1)
    begin
      if (!Sync1)
      begin
        Cmd_rdy1      = ~(|(Mask[14:0]));
       Next_rdy1      = ~(|(Mask[14:0]));
      end
      else if (Tread1)                          // Read
      begin
       case (Csd1)                              // synopsys parallel_case
       3'h1: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:5]));
              Next_rdy1 = ~(|(Mask[14:5]));
             end
       3'h2: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:6]));
              Next_rdy1 = ~(|(Mask[14:6]));
             end
       3'h3: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:7]));
              Next_rdy1 = ~(|(Mask[14:7]));
             end
       3'h4: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:8]));
              Next_rdy1 = ~(|(Mask[14:8]));
             end
       3'h5: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:9]));
              Next_rdy1 = ~(|(Mask[14:9]));
             end
       3'h6: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:10]));
              Next_rdy1 = ~(|(Mask[14:10]));
             end
       default: begin
              Cmd_rdy1  = ~(|(Tdatardy[14:5]));
              Next_rdy1 = ~(|(Mask[14:5]));
                end
       endcase
      end
      else                                      // Write
      begin
       Cmd_rdy1  = ~(|(Tdatardy[14:2]));
       Next_rdy1 = ~(|(Mask[14:2]));
      end
    end
```

FIG.5E

```
// ****************
// Generate next bit stream by concatenating old bit stream with new bit stream.
// Bit stream includes a right shift of one bit.
// ****************

  always @(Tdatardy or Csd1 or Busy1 or Sync1 or Tread1 or Mask1)
  begin
    if (Sync1 && Tread1)                        // Read
    begin
    casex (Csd1)                                // synopsys parallel_case
    3'h1: begin
            Next_tdatardy1 = {6'h0,Busy1,Tdatardy[4:1]};
            Next_mask1     = {5'h0,Mask1,Tdatardy[4:1]};
          end
    3'h2: begin
            Next_tdatardy1 = {5'h0,Busy1,Tdatardy[5:1]};
            Next_mask1     = {4'h0,Mask1,Tdatardy[5:1]};
          end
    3'h3: begin
            Next_tdatardy1 = {4'h0,Busy1,Tdatardy[6:1]};
            Next_mask1     = {3'h0,Mask1,Tdatardy[6:1]};
          end
    3'h4: begin
            Next_tdatardy1 = {3'h0,Busy1,Tdatardy[7:1]};
            Next_mask1     = {2'h0,Mask1,Tdatardy[7:1]};
          end
    3'h5: begin
            Next_tdatardy1 = {2'h0,Busy1,Tdatardy[8:1]};
            Next_mask1     = {1'h0,Mask1,Tdatardy[8:1]};
          end
    3'h6: begin
            Next_tdatardy1 = {1'h0,Busy1,Tdatardy[9:1]};
            Next_mask1     = {Mask1,Tdatardy[9:1]};
          end
    default: begin
            Next_tdatardy1 = 14'bxx_xxxx_xxxx_xxxx;
            Next_mask1     = 14'bxx_xxxx_xxxx_xxxx:
          end
    endcase
   end
  else if (Sync1 && !Tread1)                    // Write
  begin
    Next_tdatardy1 = {9'h00,Busy1,Tdatardy[1]};
    Next_mask1     = {8'h00,Mask1,Tdatardy[1]};
   end
  else                                          // Async
  begin
     Next_tdatardy1 = 14'h0000;
     Next_mask1     = 14'h0000;
   end

end
```

FIG.5F

```
// ****************
// Determine if new access can issue R/W command.
// ****************

    always @ (Tdatardy or Mask or Csd0 or Sync0 or Tread0)
    begin
      if (!Sync0)
      begin
        Cmd_rdy0        = ~(|(Mask[14:0]));
        Next_rdy0       = ~(|(Mask[14:0]));
      end
      else if (Tread0)                          // Read
      begin
       case (Csd0)                              // synopsys parallel_case
        3'h1: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:5]));
                Next_rdy0 = ~(|(Mask[14:5])):
              end
        3'h2: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:6]));
                Next_rdy0 = ~(|(Mask[14:6])):
              end
        3'h3: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:7]));
                Next_rdy0 = ~(|(Mask[14:7])):
              end
        3'h4: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:8]));
                Next_rdy0 = ~(|(Mask[14:8])):
              end
        3'h5: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:9]));
                Next_rdy0 = ~(|(Mask[14:9])):
              end
        3'h6: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:10]));
                Next_rdy0 = ~(|(Mask[14:10])):
              end
        default: begin
                Cmd_rdy0  = ~(|(Tdatardy[14:5]));
                Next_rdy0 = ~(|(Mask[14:5])):
                 end
        endcase
      end
      else                                      // Write
      begin
         Cmd_rdy0  = ~(|(Tdatardy[14:2]));
         Next_rdy0 = ~(|(Mask[14:2])):
      end
    end
```

FIG.5G